Sampled Waveform Data

Booster beam loss monitors

Tue, May 16, 2000

Booster beam loss monitors are digitized at about 10KHz throughout the 33 ms acceleration cycle. It is useful to sample all such loss monitors at a specified time. With all loss monitor waveforms operating at the same rate and triggered by the same event plus delay, it is only necessary to sample a particular word within each waveform buffer.

If a single user scheme is employed, one can specify via a dummy channel what the offset should be within the Booster cycle. The waveform buffer can then be sampled each 15Hz cycle to update another channel reading. A request for the readings of all such sample channels would be necessary to monitor sampled beam losses around the Booster ring. There may be 60–70 such loss monitors in all divided up among perhaps 10 nodes.

If a multiple user approach is needed, one would have to include in a data request an index that could refer to a given point on the waveforms. For the `RETDAT` protocol, the only choice is the offset word, as the `SSDN` is fixed in the database. Each requester could specify a different offset word. To add this logic to the `RETDAT` support in SRMs, on could use a memory access listype number with offsetting enabled, in which the offset word would be the byte offset into the waveform pointed to by the memory address in the `SSDN`.

Another approach may use a new listype whose ident includes a channel number and an offset that could be a word index number. In this case, the support code would need to derive the waveform address from the channel number. One way to do this is to search the `CINFO` system table entries for a match on the channel number. Another approach is to obtain the waveform address from the analog control field, assuming that the channel has no such analog control.

To support an Acnet console page `B36`, it can be changed to accept a time delay in ms units. The application can convert this into a suitable offset value to be used with the data requests for the beam loss waveform samples. This scheme allows multiple users of `B36` to use independent time delays in sampling the waveforms. (The current scheme actually sets a clock event to establish the time of sampling across multiple front ends, which implies single user approach and also consumes a clock event for a software purpose.)

In the Classic protocol there is no offset word. But there is the ability to define idents of whatever length is needed to include the additional required parameters. One could define a listype whose ident included both a channel number and an offset, or index number. The `RETDAT` offset logic would allow the offset word to be added to the two words (channel number, index number) to pick out the desired sample delay.

Data stream access listype

Define a listype that uses an ident that includes a data stream index and a byte offset. The targeted data would be the most recently-recorded data stream record, and the offset would determine where the data access began within the record.

Array device approach

Currently, `RETDAT` does not support byte offsets to analog channel-based data. If an `SSDN` using listype 0, which means analog readings, is used, and the offset flag is 0, an error code is returned. But if the channel can be related to an array of data, as in a waveform, the support could be modified to allow for byte offsets. The way to relate a channel to a waveform is via the `CINFO` table, which is used by `FTPMAN` to obtain the waveform memory base address

and the address of the digitizer interface registers. So RETDAT can be modified to search CINFO for a match on the channel number. A match can yield the base address of the waveform, to which the byte offset can be added to get the address of the data item sought. In this case, if the number of bytes requested is larger than 2, but a multiple of the data item size, the indicated number of bytes can be read.

In some cases, a listype 0 access using the channel-based ident should result in accessing the reading field of the ADATA. In other cases, there is no such reading word that has meaning. The same SSDN can mean to access the reading field, or it can mean to access the first word (offset=0) of the waveform array. To resolve this ambiguity, assume that a 2-byte request with a zero offset value should result in access to the reading word; otherwise, it accesses part of the waveform array. This would rule out access to the first word alone of the waveform, although accessing more than one word using a zero offset would obtain the first words of the array.

The scheme described will not apply to the kHz waveform data, in which successive digitized values are separated by 128 bytes in the hardware circular buffer. It is difficult to know where to start.

Using FTPMAN to access Swift or Quicker digitized data, it is necessary to distinguish two types of support. One is the current support that uses a message queue to communicate with the SWFT local application. The other support would access the hardware waveform buffer directly. It would return values for the delay and the digitize rate as given by the registers. But it can permit waiting for the event used in the request. In this second mode, the hardware would operate in the auto-trigger mode, such that the waveforms would be measured at 15Hz. When the given clock event is noticed to have occurred in the present cycle, the contents of the hardware buffer are copied out for subsequent access by the FTPMAN client.

15Hz waveform access

Acnet does not support reliable 15Hz access to waveform data, either via FTPMAN or by RETDAT. One scheme that has been considered to get around the limitation using RETDAT is to request that multiple waveforms be delivered at 7.5Hz. An application program should be able to collect data at 7.5Hz reliably. Included with the waveform data must be some time stamp information so that the client knows on what cycle each waveform was measured. One way to do this is via data streams. A local application would have to capture the waveforms into a data stream that can hold at least 2 waveforms for access via one of the listypes designed to support data stream access. This would need to be done at least every Booster reset cycle of interest, if not at 15Hz. An active request for such data stream waveforms would be fulfilled with 0–2 waveforms. Naturally, as a special application would need to assimilate such reply data.